A tangible-augmented concept inventory to identify novices' misconceptions in programming

Henry, Julie; Magis, Tom; Clarinval, Antoine; Vanderose, Benoit; Dumas, Bruno

Published in:
15th International Conference on Computer Science and Education (ICCSE 2020)

DOI:
10.1109/iccse49874.2020.9201806

Publication date:
2020

Document Version
Early version, also known as pre-print

A Tangible-Augmented Concept Inventory to Identify Novices' Misconceptions in Programming

Henry Julie
Namur Digital Institute
University of Namur
Namur, Belgium
julie.henry@unamur.be

Magis Tom
Namur Digital Institute
University of Namur
Namur, Belgium

Clarinval Antoine
Namur Digital Institute
University of Namur
Namur, Belgium
antoine.clarinval@unamur.be

Vanderose Benoit
Namur Digital Institute
University of Namur
Namur, Belgium
benoit.vanderose@unamur.be

Dumas Bruno
Namur Digital Institute
University of Namur
Namur, Belgium
bruno.dumas@unamur.be

***Abstract*—Introductory programming courses are a challenge for any teacher and a barrier for undergraduate students. A tangible-augmented concept inventory (TACI) is developed according to a design-oriented methodology in close collaboration with future users (students, teachers). It aims to help teachers to become aware of the misconceptions their students have and to rectify them appropriately. Tests conducted with 8 experts and 9 students show promising results and lead to improvement of the TACI.**

***Index Terms*—Computer science education, Concept inventory, Tangible interaction, Design-oriented research, Misconceptions**

I. Introduction

Learning programming is considered difficult for novices [1] of any age. Introductory programming courses are a challenge for any teacher and often result in a quite high failure and drop-out rates among undergraduate students. More and more learning resources are being developed to assist in the exploration of basic programming concepts and computational thinking [2]–[4]. In particular, tangible interaction is a preferred approach to working with children [5]. However, it is rarely used when teaching young adults [6].

Many factors come into play in explaining the difficulties experienced by beginner-level programmers [7]. In particular, one of them can help teachers to improve their teaching: the misconceptions [8] that students hold beforehand. These misconceptions hinder their progress and may discourage them from continuing their learning. They can be identified using a *concept inventory* (CI), which is a "research-based multiple-choice test that seeks to measure a student's knowledge of a set of concepts while also capturing conceptions and misconceptions they may have about the topic under consideration" [9].

Thus, relying on the previously documented success of tangible devices and CI about basic programming concepts [10], we argue that a tangible-augmented CI (TACI) could help teachers to become aware of the misconceptions their students have and to rectify them appropriately during subsequent practical work sessions [11]. The students as well, through the manipulation of the TACI, should be able to perceive their misunderstandings and to correct them. In order for such a device to best meet the needs, its development must be conducted in close collaboration with future users, i.e. students, but also with teachers and other education experts.

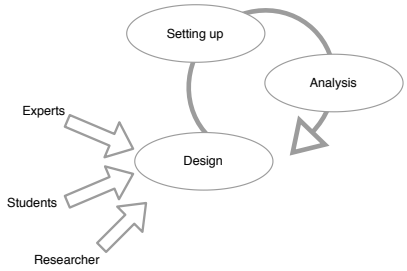

Fig. 1. Design-oriented research methodology

II. TACI Design

The TACI consists of a set of programming problems that the user must solve and which allow measuring his/her understanding of basic programming concepts. Rather than answering a multiple-choice question, the user freely constructs his/her answer.

The TACI is developed following a methodology inspired by *design-oriented research* [12]. It is an iterative process that articulates phases of *design*, *set up* with different audiences, and *analysis* of the collected data. Thus, development is a collaboration between the researcher developing the TACI and experts, but also between the researcher and future users, namely students, and education professionals (teachers, tutors, etc.) (Figure 1).

Two aspects of the TACI are distinguished: the didactic aspect, i.e. the misconceptions to be identified and how to identify them, and the human-computer interaction (HCI)

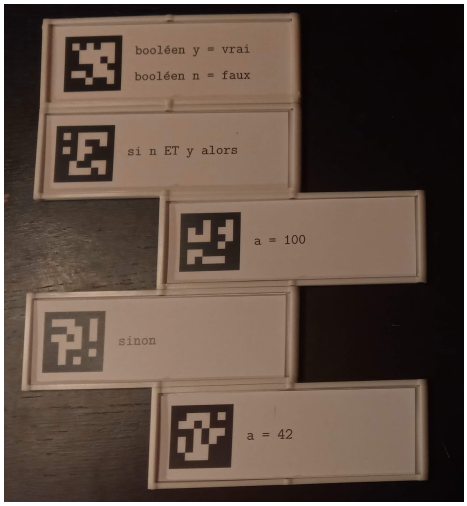

Fig. 2. A program written in blocks

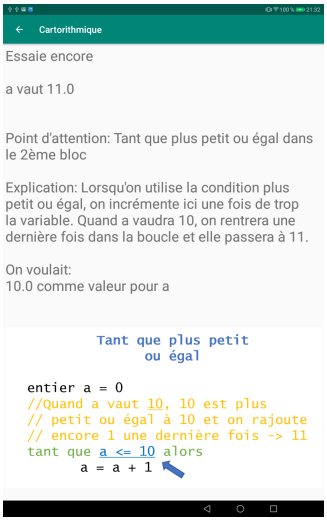

Fig. 3. Feedback screen

aspect, i.e. the shape of the TACI and the interactions with it. Experts were invited to collaborate on both aspects.

A. Didactical Aspect

Through the TACI, the user builds a program answering a short problem focused on the manipulation of a basic programming concept. Here, the concepts covered are the variable, the conditional structure, and the loop. Based on [10], several problems have been foreseen, allowing to identify major misconceptions such as the reading direction of a variable assignment, the number of iterations of a loop according to its condition statement, or the understanding of Boolean algebra within a condition.

A problem is composed of a statement and of a specific set of blocks to assemble to build the program answering the statement (Figure 2). Each block represents an instruction written in pseudo-code. These instructions can be incorrect, illustrating some of the misconceptions that learners may have. The solution proposed must be evaluated and the result of this evaluation must be made explicit. Thus, the following information must be provided: the value of the variables used in the program when it is executed; if the solution is incorrect, the line impacted and the error that occurred; a theoretical explanation of the concept correcting the misconceptions (Figure 3).

B. HCI Aspect

The choice of a programming language working with tangible blocks takes its inspiration from the numerous existing devices [13]–[15]. The blocks have been printed in 3D and are magnetically assembled one under the other, for a reading of the program from top to bottom. The blocks can be assembled in several positions to support indentation (Figure 2). Each block represents one instruction printed on a strip of paper attached to it. Each instruction has a unique tag, and thus can be scanned and recognized.

Fig. 4. Home screen of the application

Besides the set of tangible blocks, the TACI includes an application running on a tablet or a smartphone. It is composed of three screens: a home screen, a screen for scanning, and a screen displaying feedback to the user. Touching one screen allows moving to the next one, and a backspace button allows going back to the previous screen. First, the home screen explains how the TACI works (Figure 4). Second, the scanning screen returns what is captured by the tablet/smartphone's camera. The number of recognized blocks (and thus, recognized instructions) is indicated, allowing the user to correct the scan if necessary (Figure 5). Third, the feedback screen displays the feedback associated to the scanned program (Figure 3).

III. TACI Set Up

The first design iterations of the TACI carried out in collaboration with experts lead to a functional prototype. Two test phases were then organized. The first phase consisted in presenting the TACI to 8 computer science education experts who teach programming courses or practical sessions and who would potentially use the TACI with their students. The second phase involved 9 students following an introductory programming course in their first year of university. The objective of these test phases was to obtain feedback on the user experience, but also on the potential of the TACI to be used in a teaching/learning context.

A. Formative Test with Teaching Staff

The participants were asked to solve three problems using a set of blocks containing pseudo-code instructions specific to each problem. **Problem 1** involves a set of 12 blocks and tests the understanding of the concept of variable. It consists in writing a program which result is an exchange of value between two variables. **Problem 2** involves a set of 8 blocks and tests the understanding of the concept of conditional structure. It consists in writing a program which result is the assignment of a defined value to a variable when a condition involving Boolean algebra is present. **Problem 3** involves a set of 7 blocks and tests the understanding of the concept of loop. It consists in writing a program with a defined number of loop iterations.

Ethnographic observations were carried and the User Experience Questionnaire (UEQ) [16] was completed by all participants.

Six experts (75%) found problem 2 too difficult for novices. In addition, the pseudo-code used to write the instructions confused some experts. Two experts (25%) were surprised that typing was explicit ("integer a = 2"). Six experts omitted the "end if" block. Two experts said they had difficulties to have a global view of all the blocks available in a set.

Regarding the application, some ergonomic problems have been reported for the scanning screen. The position of the button to launch the scan made the tablet difficult to handle. In addition, the success of the scan and the recognition of the instructions were not perceived by the experts. The feedback presented following the evaluation of a solution was also commented. Indentation problems, which generate a warning, should be reported as an error instead. The red color used to emphasize some information was considered too aggressive. Finally, the general layout of the feedback, the different fonts used, and the insufficient size of the characters made reading difficult. Most of the experts did not read all of the information, being more attracted by the images than by the text.

Overall, the TACI was well-received by the experts. Moreover, it was considered rather easy to use and learn: an error observed in the interaction with the TACI is not reproduced afterwards. To support these results, a UEQ user experience questionnaire was completed by each of the experts at the end of the session. Six criteria were evaluated on a 7-point Likert scale ranging from -3 to 3 (Figure 6): attractiveness, perspicuity, efficiency, dependability, stimulation, and novelty.

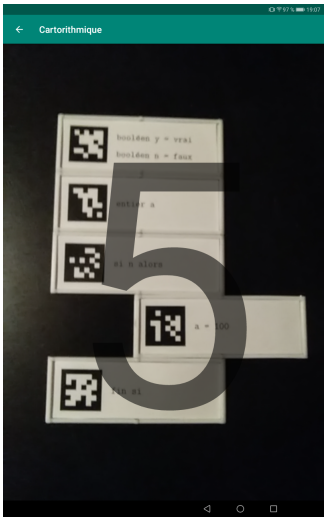

Fig. 5. Blocks recognized by the application

The data collected through this test phase led to the TACI version presented in this article. The "end if" and "end while" statements have been removed. The indentation of blocks has been made mandatory. The button launching the scan has been replaced by a simple tap on the touch screen. Finally, all messages sent to the user have been revised to improve their readability.

B. Early Validation with Students

The nine students who participated in the second test phase have been taking an introductory university course in programming for one semester. These students came from two different majors: four of them studied Computer Science (INFO) and five studied Business Engineering (INGE). They were randomly selected among the 140 students following the introductory course. The entire experiment was video recorded from two perspectives: a global perspective capturing the students' interactions with the blocks and with the researcher, and a more focused perspective on the students' interactions with the application running on a tablet. The resulting videos were annotated and the full dialogues were transcribed.

Only problem 1 was proposed to the students. In order to measure the influence of the pseudo-code, and taking into account the observations made by the experts, the instructions were translated into natural language (Figure 7). The two sets of blocks (blocks in pseudo-code and blocks in natural language) were proposed simultaneously. The students were free to choose the one they wanted to start with.

Six students (5 INGE and 1 INFO - 66%) preferred to start the test with the instructions written in natural language. One student chose to start with the instructions in pseudo-code because he was afraid that he would misunderstand some of the nuances when reading natural language. Two of the six students who were first confronted to natural language were

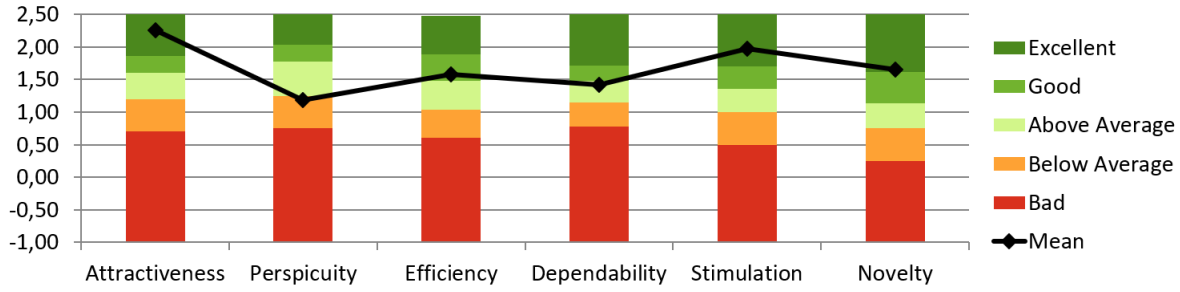

Fig. 6. UEQ Results

Pseudo-code: *integer a = 3*
Natural language: *I declare a variable of integer type, name it "a" and initialize it to 3*

Fig. 7. Assign a variable in two languages

unable to solve the problem. For one of them (1 INFO), the use of pseudo-code did not help to propose a solution either. Three of these six students chose the wrong block describing a variable assignment, showing that they were reversing the reading direction of the = symbol.

Natural language was considered more difficult by 8 students (88%), in particular because of the length of the texts. The pseudo-code is, according to the students, closer to what they know (i.e. the Python language). However, one student (1 INGE) said that he better understood the instruction in its working thanks to natural language, and three students (3 INFO) did not understand typing and thus ignored it. The students admitted that results may have differed, had they manipulated the TACI before starting the introductory programming course. One student suggested coloring the blocks by instruction type.

Two students experienced difficulties with the application to return to a previous screen and to scan the blocks. These difficulties were easily overcome.

Regarding the feedback received by the application, three students (3 INFO - 33%) could not locate the error because they did not understand the information on the incorrect block number. One explained that he considers that a block represents several instructions, and not only one. Four students (44%) did not read all the information displayed after the scan and focused instead on the information on the error location.

One student (1 INGE - 11%) managed to solve the problem in both pseudo-code and natural language in one try. One student (1 INFO) experimented with the TACI before trying to solve the problem to see the type of feedback provided after the scan.

Overall, the TACI and its potential use in the practical work sessions of their course was well-received by the students. Nonetheless, one student (1 INGE) mentioned that the playfulness of the TACI could be a risk, as it opposes to the serious nature of learning.

IV. TACI Analysis

Regarding the first test phase, the results of the UEQ (Figure 6) were compared with the values of a standard dataset[1], following UEQ analysis practice. It emerged that two aspects of the TACI need to be improved, namely perspicuity and dependability.

During the second test phase, interactions with the application appeared to have improved. More work needs to be done on the feedback because some students did not read all the information displayed and therefore missed out on some of the learning, such as the awareness of their misconceptions. On the other hand, the use of the TACI allowed understanding the difficulties that students have in reading instructions in natural language and, in turn, in understanding basic programming concepts. Their preference for pseudo-code is explained by its proximity to the Python programming language and by the routines they have implemented during their learning. It appears that they are able to write an assignment, without understanding how it works, by retaining a pattern to reproduce.

V. Conclusion

A tangible-augmented concept inventory (TACI) is developed following an iterative methodology emphasizing collaboration between experts, students, and teachers. It aims to help introductory programming course teachers to become aware of the misconceptions their undergraduate students have and to rectify them appropriately.

Although the results of the first iterations are promising, there is still work to be done. First, on the feedback presented by the application, as the current TACI does not help students to become aware of their misconceptions. Second, on the implementation of additional problems to allow identifying a majority of the misconceptions. Finally, additional tests should be conducted with real novices to observe their behavior toward instruction languages.

[1]https://www.ueq-online.org/Data from 452 studies involving 20,190 people and relating to the analysis of software, web pages, social media, web shops, etc.

REFERENCES

[1] Robins, A., Rountree, J., & Rountree, N. (2003). “Learning and teaching programming: A review and discussion”. Computer science education, 13(2), 137-172.

[2] Kelleher, C., & Pausch, R. (2005). “Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers”. ACM Computing Surveys (CSUR), 37(2), 83-137.

[3] Gross, P., & Powers, K. (2005, October). “Evaluating assessments of novice programming environments”. In Proceedings of the first international workshop on Computing education research (pp. 99-110).

[4] Yu, J., Roque, R. (2018, June). “A survey of computational kits for young children””. In Proceedings of the 17th ACM conference on interaction design and children (pp. 289-299).

[5] Shaer, O., & Hornecker, E. (2010). “Tangible user interfaces: past, present, and future directions”. Foundations and Trends® in Human–Computer Interaction, 3(1–2), 4-137.

[6] Henry, J., Dumas, B., & Bodart, A. (2018, October). “Programmation tangible pour les enfants: analyse de l’existant, classification et opportunités”.

[7] Lahtinen, E., Ala-Mutka, K., Järvinen, H. M. (2005). “A study of the difficulties of novice programmers”. ACM sigcse bulletin, 37(3), 14-18.

[8] Sorva, J., Karavirta, V., & Malmi, L. (2013). “A review of generic program visualization systems for introductory programming education”. ACM Transactions on Computing Education (TOCE), 13(4), 1-64.

[9] Wittie, L., Kurdia, A., & Huggard, M. (2017, October). “Developing a concept inventory for computer science 2”. In 2017 IEEE Frontiers in Education Conference (FIE) (pp. 1-4). IEEE.

[10] Henry, J., & Dumas, B. (2019, October). “Towards the identification of profiles based on the understanding of programming concepts: the case of the variable”. In 2019 IEEE Frontiers in Education Conference (FIE) (pp. 1-8). IEEE.

[11] Qian, Y., Hambrusch, S., Yadav, A., Gretter, S., & Li, Y. (2020). “Teachers’ Perceptions of Student Misconceptions in Introductory Programming”. Journal of Educational Computing Research, 58(2), 364-397.

[12] Anderson, T., & Shattuck, J. (2012). “Design-based research: A decade of progress in education research?”. Educational researcher, 41(1), 16-25.

[13] Marco, J. B., Baptiste-Jessel, N., & Truillet, P. (2018, October). “TaBGO: programming with tangibles blocks”. In Proceedings of the 30th Conference on l’Interaction Homme-Machine (pp. 179-185).

[14] Horn, M. S., & Jacob, R. J. (2007, April). “Tangible programming in the classroom with tern”. In CHI’07 extended abstracts on Human factors in computing systems (pp. 1965-1970).

[15] Horn, M. S., & Jacob, R. J. (2007, February). “Designing tangible programming languages for classroom use”. In Proceedings of the 1st international conference on Tangible and embedded interaction (pp. 159-162).

[16] Schrepp, M., Hinderks, A., & Thomaschewski, J. (2017). “Design and Evaluation of a Short Version of the User Experience Questionnaire (UEQ-S)”. IJIMAI, 4(6), 103-108.